PROBLEMAS DE OPTIMIZACIÓN DINÁMICOS: ENFOQUES Y PERSPECTIVAS

Antonio D. Masegosa*, David A. Pelta**
*Facultad de Ingeniería, Universidad de Deusto, 48007, Bilbao, España.
DeustoTech-Fundación Deusto, Fundación Deusto, 48007, Bilbao, España.
IKERBASQUE, Fundación Vasca para la Ciencia, 48011, Bilbao, España.
**Departamento de Ciencias de la Computación e Inteligencia Artificial,
E.T.S. de Ingenierías Informática y de Telecomunicación
Universidad de Granada, 18014, Granada, España

ABSTRACT
When we face an optimization problem whose definition (in some aspect) changes over the time, we are in the presence of a Dynamic Optimization Problem (DOP). The aspects that can change are the objective function, the variables' domain, the appearance/disappearance of variables or constraints, etc. This paper aims at providing a first introduction to those who are interested in the topic. Concretely, we present the DOPs, the most common performance measures as well as the methods used to solve them. We also briefly describe some of the most recent reviews and comment some current challenges and research opportunities in DOPs.

KEYWORDS: Dynamic Optimization Problems, Dynamic Environments, Metaheuristics, Problem Generators, Performance Measures, Tutorial

MSC: 90C59

RESUMEN
Cuando nos enfrentamos a un problema de optimización cuya definición (en algún aspecto) cambia a lo largo del tiempo, estamos en presencia de un problema de optimización dinámico (POD). Los aspectos cambiantes pueden ser la función objetivo, el dominio de las variables, la aparición/desaparición de variables o restricciones, etc. Este artículo pretende servir como primera introducción para los interesados en el tema. Se presentan los PODs, las medidas de rendimiento y los métodos utilizados para resolverlos. Se describen brevemente algunas revisiones recientes y se comentan desafíos y oportunidades de investigación en el ámbito de los PODs.

1. INTRODUCCIÓN

La Inteligencia Computacional (IC) y la Investigación de Operaciones (IO) surgieron para dar respuesta a problemas complejos que surgen en el contexto socio-tecnológico actual. En la interfaz entre ambas áreas encontramos los problemas de optimización. Algunos ejemplos de estos problemas son obtener la mejor ruta para el reparto de mercancías, la mejor asignación de tareas a equipos o decidir cuáles son las mejores localizaciones para instalar servicios.

Para poder resolver estos problemas en un ordenador, tanto IC como IO, procuran encontrar modelizaciones adecuadas que reflejen lo más ajustadamente posible el problema "real" así como métodos computacionales para abordar su resolución.

Idealmente, estos métodos retornarán la solución óptima, pero existen diferentes razones que justifican el uso de métodos aproximados o heurísticos para abordar la resolución de estos problemas de optimización, lo cual, plantea además dos cuestiones importantes: ¿cómo medir el rendimiento de los métodos?, y ¿cómo compararlos entre sí?.

Estos tres aspectos, los problemas, los métodos y las medidas, hacen que la investigación en la resolución de problemas de optimización mediante métodos aproximados sea una tarea compleja.

Por otro lado, existen dos factores que en escenarios reales, pueden y deben considerarse: la incertidumbre o vaguedad en los datos del problema, y el dinamismo. En otras palabras, el mundo real no es estático ni

está definido de forma precisa.
No abordaremos aquí el problema de modelización y manipulación de la incertidumbre, sobre lo cual el lector interesado puede consultar [15, 12]. El foco lo pondremos sobre el dinamismo. Consideremos los siguientes ejemplos:

- Se desea resolver el problema del viajante de comercio donde los arcos entre ciudades se encuentran etiquetados con información de tráfico. Sería un error asumir que dicha información es "constante" puesto que las condiciones de tráfico varían de forma dinámica.
- Se desea obtener las rutas de reparto para un conjunto de vehículos de una empresa de logística. Sin embargo, mientras los vehículos se encuentran en ruta, la empresa recibe pedidos de recogida de paquetes que deben ser atendidos. Esto implica re-planificar las rutas.
- Supongamos un problema de inversión en bolsa donde hay que distribuir un presupuesto fijo entre un conjunto de valores. Si no consideramos la posibilidad de redistribución de la asignación, nuestra solución será poco realista.

Estos ejemplos asumen que el problema varía a lo largo del tiempo y esta variación puede darse en la función objetivo, en el dominio de las variables o incluso en el número de éstas. Estamos por tanto, frente a problemas de optimización dinámicos (PODs).
Así como se identificaron tres cuestiones relevantes en la resolución de problemas de optimización "estáticos", se plantea como objetivo de este trabajo discutir dichas cuestiones en el contexto de los PODs. Se pretende que este trabajo sirva de guía introductoria para que, posteriormente, el lector interesado en el tema pueda abordar cuestiones más complejas.
Para ello, el artículo se organiza de la siguiente manera. En la Sección 2. se presentan los conceptos básicos de los PODs. Las medidas de rendimiento y las estrategias de resolución más usuales aparecen en las Secciones 3. y 4.. Finalmente, en la Sección 5. se describen algunas revisiones recientes publicadas en los últimos 5 años y se plantean una serie de perspectivas que pueden servir de base a futuras investigaciones en el ámbito de los PODs.

2. PROBLEMAS DE OPTIMIZACIÓN DINÁMICOS

Un POD se puede definir de la siguiente manera. Siendo $\Omega^{(t)} \subseteq \mathbb{R}^n$ el espacio de búsqueda:

$$\min f^{(t)}(\mathbf{x}) \tag{1}$$

donde $x \in \Omega$ y t representa el tiempo. Para la modelización, implementación y resolución de un POD se considera t como una magnitud discreta ($t \in \mathbb{N}$), asociada generalmente con el número de evaluaciones de la función objetivo del problema. Debe notarse que cada elemento del problema depende de t, lo cual significa que en cada instante de tiempo puede variar la función objetivo, el espacio de búsqueda o ambos a la vez. En caso de considerar elementos adicionales para el problema, como por ejemplo un conjunto de restricciones, estos también podrían depender de t. Aparecen aquí algunas suposiciones básicas:

1. Durante el período entre t y $t+1$, el problema permanece estático.
2. Los cambios son graduales (no bruscos). Dada una solución $\mathbf{x}$, se espera que su "calidad" entre dos instantes de tiempo consecutivos no varíe significativamente.
3. Los cambios son informados o se pueden detectar.
4. La frecuencia y la severidad de los cambios son factores relevantes. En problemas reales pueden no ser conocidos, pero en escenarios de prueba artificiales son parámetros indispensables.
5. El período de tiempo entre cambios no es tan largo como para re-optimizar "desde cero". Es decir, no se debería considerar al POD simplemente como un encadenamiento de problemas estáticos (aunque esta aproximación no debe ser descartada por completo).

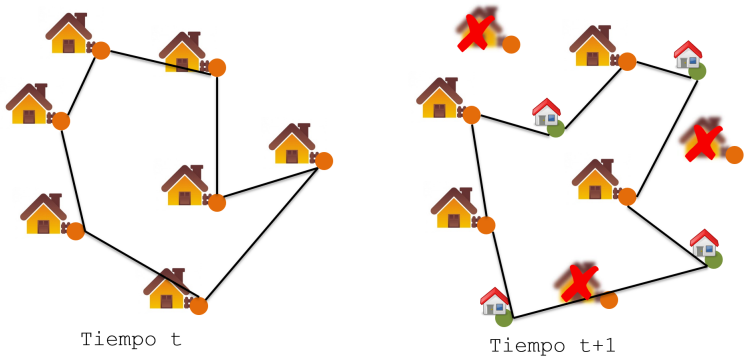

Figura 1: Ejemplo del problema del viajante dinámico: las ciudades pueden aparecer y desaparecer.

En este sentido, la resolución de un POD requiere re-optimizar lo más rápido posible cuando ocurre un cambio.
En la Fig. 1 se puede observar un ejemplo del problema del viajante de comercio dinámico, donde, entre dos instantes consecutivos de tiempo, aparecen y desaparecen ciudades (ha cambiado el tamaño del problema).

Una problemática que suele encontrar el investigador que quiere abordar un POD o introducirse en el tema, es la cuestión de dónde obtener "problemas de ejemplo" o instancias. En el caso de problemas estáticos existen repositorios como OR-LIB [4] o TSPLIB [24] que proveen una amplia variedad de instancias o ejemplos de problemas de optimización.
En el caso de los PODs la situación no es así. Usualmente los investigadores recurren a generadores de problemas que buscan simular las características de los PODs reales en un entorno controlado, en el que poder definir diferentes tipos de dinamismo mediante el ajuste de diversos factores como, por ejemplo, la frecuencia o la severidad de los cambios. El número de generadores de PODs disponibles hoy en día es bastante extenso y abarcan prácticamente todas las variantes que podemos encontrar de estos problemas atendiendo a criterios como el tipo de codificación de las soluciones, el número de objetivos, restricciones, factores que cambian, predictibilidad de los cambios, etc. El lector interesado puede consultar los artículos de revisión y libros comentados en la Sección 5., para un análisis completo de los generadores de PODs propuestos hasta ahora.
A continuación describiremos brevemente los generadores más usados en la literatura.

2.1. GENERADORES DE PROBLEMAS

- **Esquema genérico**. La forma más simple de generar PODs de prueba es "enlazar" un conjunto de problemas estáticos donde se han variado los datos. Así es como se pueden construir versiones dinámicas de problemas bien definidos como el mencionado problema del viajante de comercio, el problema de la mochila o el problema de localización.

- **Problema de los Picos Móviles (MPB: Moving Peaks Benchmark) [6].** Fue uno de los primeros benchmarks propuestos para PODs y probablemente uno de los más utilizados en este campo. Representa un paisaje multi-dimensional donde un conjunto de picos con forma cónica varían su posición, altura y anchura ligeramente, dentro de un espacio de soluciones continuo, cada vez que se produce un cambio. El generador permite modular diferentes tipos de dinamismo y de complejidad a través de parámetros como la dimensión, el número de picos, la frecuencia y la severidad de los cambios o el grado de correlación que siguen los movimientos de los picos. La función que define los picos también se puede modificar.

- **Generador genérico de problemas dinámicos (GDBG: Generalized Dynamic Benchmark Generator") [17].** El GDBG pretende ser un generador de problemas tanto binarios, continuos como combinatorios. Provee 6 tipos de dinamismo que son: cambios aleatorios, cambios pequeños y grandes, cambios caóticos, recurrentes y recurrentes con ruido. Es discutible si cada tipo de cambio se corresponde o no con algún tipo de dinamismo en el mundo real. Este generador se utilizó para

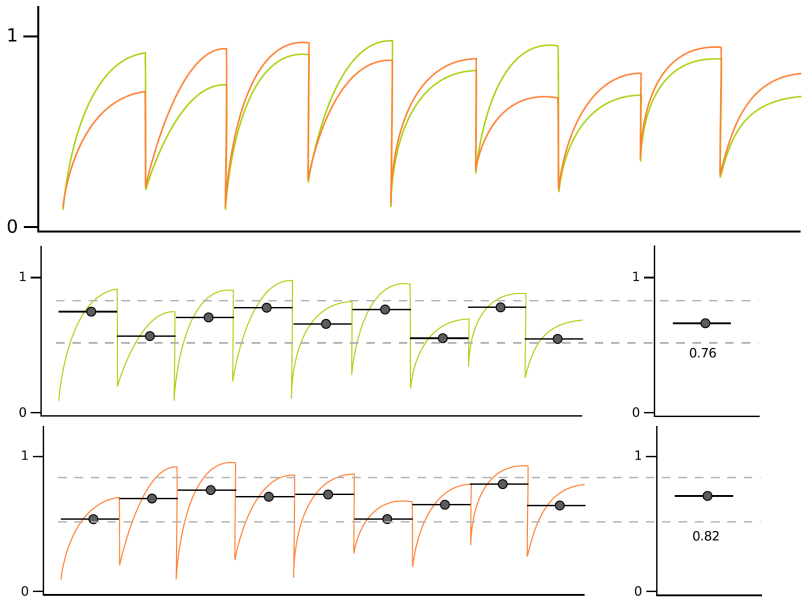

Figura 2: Rendimiento de dos algoritmos sobre un POD. La cuestión que se plantea es ¿cuál es mejor?

una competición de algoritmos para POD en el congreso IEEE-WCCI 2012. Las instrucciones de la competición, así como el código del generador se pueden acceder desde `http://people.brunel.ac.uk/~csstssy/ECDOP-Competition12.html`

- **Generador de problema binarios dinámicos XOR (XOR-DPG: XOR Dynamic Problem Generator) [30].** Este generador permite transformar cualquier problema de optimización binario estático en uno dinámico. El mecanismo es muy sencillo, en el proceso de evaluación de las soluciones se añade un paso intermedio en el que a la solución en cuestión se le aplica una determinada máscara mediante un operador O-excluivo (XOR). La cadena de bits resultante de la operación XOR entre la solución y la máscara, es la que finalmente se evalúa, estableciendo así el fitness de la solución en cuestión. Esta máscara cambia en cada etapa del problema creando así dinamismo en el problema, pudiéndose ajustar su frecuencia y severidad a través de los cambios en la máscara. Al igual que ocurre con el GDBG, no está claro que los cambios inducidos por este mecanismo sean similares a los que tienen lugar en los problemas reales.

- **Generador de Problema Dinámicos Multi-objetivo (DMOPG: Dinamic Multi-Objective Problem Generator)[14].** Aunque ya existían en la literatura algunos generadores de PODs multi-objetivo [21], este generador propuesto recientemente por Jiang and Yang, ofrece un entorno bastante completo en el que controlar la variación a lo largo del tiempo de importantes características de este tipo de problemas como la frontera pareto-optimal, el conjunto pareto-optimal, la interdependencia entre variables, la uni-modalidad o multi-modalidad del problema, el tipo de dinamismo o el nivel de aleatoriedad de los cambios. La gran versatilidad del generador lo hace muy útil para hacer estudios teóricos o empíricos sobre PODs multi-objetivo.

3. MEDIDAS DE RENDIMIENTO

El problema de comparar el rendimiento de dos algoritmos sobre un POD queda reflejado en la Fig. 2. Para medir el rendimiento de un algoritmo sobre un POD, usualmente se toman una serie de medidas puntuales, por ejemplo antes de cada cambio, y luego se promedian para condensar toda la ejecución en un único valor [1]. Este esquema se muestra en las dos imágenes inferiores de la Fig. 2.

[1]Naturalmente, si los algoritmos se basan en decisiones aleatorias (como las metaheurísticas), entonces se deben realizar varias ejecuciones de cada algoritmo en cada problema.

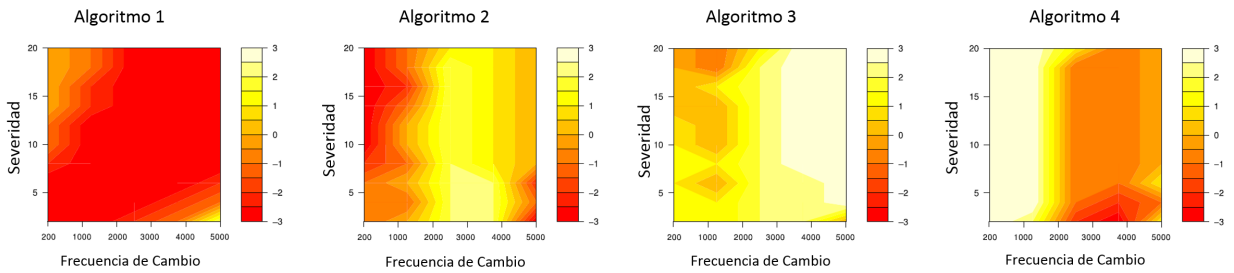

Figura 3: Resultados obtenidos mediante la técnica SRCS. Dado un algoritmo, se construye un ranking que indica, en cada escenario, a cuantos algoritmos "le gana" y con cuantos "pierde". Este ranking se transforma en una escala de color. Colores más claros indican que el algoritmo en cuestión es mejor que los otros en el escenario considerado.

Si se dispone del óptimo para cada instante de tiempo o algún valor de referencia, entonces se puede calcular el denominado "error fuera de línea":

$$error = \frac{1}{T}\sum_{t=1}^{T}\left(f^t(valor\,ref) - f^t(mejor\,sol)\right) \tag{2}$$

donde $mejor\,sol$ es la mejor solución encontrada en el período analizado y $valor\,ref$ es el óptimo para dicho período.
En otro caso, se podría considerar el rendimiento o performance directamente como:

$$performance = \frac{1}{T}\sum_{t=1}^{T} f^t(mejor\,sol) \tag{3}$$

Ahora bien, la comparación del rendimiento de diferentes algoritmos es una tarea básica en el ámbito de la optimización. Esto es una tarea compleja cuando se realiza sobre problemas "estáticos", donde el investigador considera varios algoritmos, con diferentes conjuntos de parámetros, que se ejecutan sobre varios problemas de prueba. La situación es aún más compleja en PODs puesto que se deben considerar todas las configuraciones asociadas al dinamismo. Entre otros, se requiere definir, tipo, cantidad, severidad y frecuencia de los cambios. Así, considerando simplemente dos niveles por cada factor, tendríamos 2^4 escenarios de dinamismo a considerar (dejando fija la dimensión del problema).
Si disponemos de tres algoritmos que se deben evaluar en 2^4 escenarios diferentes, está claro que el análisis de resultados no es trivial. Más aún si se quiere evaluar, por ejemplo, el impacto de la severidad de los cambios para los diferentes algoritmos o intentar entender en que situación conviene aplicar un algoritmo u otro (menos severidad y mayor frecuencia, o mucha severidad y poca frecuencia).
Esta situación da lugar al uso de tablas de gran tamaño o secuencias de gráficos que, en ocasiones, resultan difíciles de entender. Para aliviar esta situación, se propuso SRCS (Statistical Ranking Color Scheme) [10]. Es una técnica que comprime los resultados en forma gráfica, permitiendo detectar patrones de comportamiento más fácilmente que si se emplearan tablas. La construcción de dichos gráficos se basa en comparaciones estadísticas de resultados, dando lugar a imágenes como las que se muestran en la Fig. 3. Dados cuatro algoritmos, cada uno de ellos puede "ganarle" como máximo a los tres restantes o "perder" (en el peor caso) contra los tres. Este rango de $[-3, 3]$ se representa mediante una escala de colores. Cuanto más claro el color, mejor el algoritmo. De esta manera, se puede observar que el algoritmo 1 es prácticamente el peor algoritmo en todos los escenarios considerados (combinaciones de severidad de cambio, en el eje Y, y frecuencia de cambio, en el eje X). Si se observa el algoritmo 4, el color se oscurece gradualmente a medida que aumenta la frecuencia del cambio. Esto indica que este algoritmo es el mejor cuando el problema cambia lentamente y además, la severidad no parece tener gran influencia. Deducir este tipo de información a partir de tablas es un proceso mucho más complejo. Esta técnica se ha utilizado para comparar 8 algoritmos sobre más de 50 escenarios diferentes en [2].

El lector interesado en profundizar en estos aspectos puede consultar [5, 20], donde se discute el rol de diferentes medidas de rendimiento, y donde se presentan herramientas y "benchmarks" para evaluar el

rendimiento de algoritmos, respectivamente. Un buen punto de partida es la sección correspondiente a las medidas en [8].

4. MÉTODOS DE RESOLUCIÓN DE PODS

En los problemas de optimización "estáticos", el objetivo es encontrar el óptimo global o bien, obtener la mejor solución posible dados ciertos recursos (tiempo, memoria, etc.). Sin embargo, el objetivo cuando se abordan los PODs es rastrear el óptimo a lo largo de los diferentes cambios. Además, estos cambios se producen de forma gradual, por lo que la información obtenida en la etapa de resolución actual puede ayudar a hacer más eficiente la búsqueda en la siguiente etapa del problema. Dadas estas características, de una forma intuitiva es razonable pensar que una población de soluciones será más efectiva a la hora de rastrear estos cambios graduales. Es por este motivo, que la mayoría de los métodos de resolución de PODs que podemos encontrar en la literatura están basados en poblaciones [21, 32]. Sin embargo, diversos estudios han mostrado que los algoritmos basados en poblaciones "convencionales', es decir, diseñados para problemas estáticos, presentan un rendimiento pobre a la hora de abordar los PODs. Una causa obvia de este rendimiento pobre es el aspecto de la convergencia. Cuando un algoritmo basado en población ha convergido, es imposible que pueda adaptarse para resolver el problema modificado.
Por este motivo, en la literatura se han propuesto diferentes mecanismos y enfoques para diseñar métodos basados en poblaciones que permitan explotar las particularidades de los PODs (gradualidad de los cambios y rastreo de los óptimos) para así resolverlos de una forma más efectiva y eficiente. A continuación ofrecemos una descripción general de los principales mecanismos y enfoques que se han propuesto en este sentido.

a) Mecanismos de detección de cambios. Para que un algoritmo pueda reaccionar cuando ocurre un cambio, éste debe ser detectado. En entornos controlados, se puede asumir que la frecuencia de los cambios es conocida. Por ejemplo, el problema cambia cada cierto número de evaluaciones de la función objetivo.

En situaciones más cercanas a la realidad es necesario detectar los cambios y, para ello, se siguen dos enfoques diferentes: mediante la reevaluación de ciertas soluciones [13], llamadas *detectores* en este contexto, o bien mediante la detección de alteraciones en el comportamiento del método [26].

b) Enfoques basados en la diversidad. Un método que ha convergido puede tener muchos problemas a la hora de rastrear la nueva posición del óptimo una vez se haya producido el cambio, ya que probablemente toda la población de soluciones esté fuera de su nueva cuenca de atracción. Una mayor diversidad en la población de soluciones, aumenta la probabilidad de que alguno de los individuos "caiga" dentro de dicha cuenca de atracción, y así pueda guiar al método hacia la nueva posición del óptimo. En este sentido podemos encontrar dos categorías de métodos: aquellos que inducen un aumento de la diversidad tras detectar un cambio [9], y aquellos que mantienen la diversidad alta a lo largo de todo el proceso de búsqueda [31]. En el último caso, potencialmente no sería necesario detectar los cambios ya que si la diversidad está bien gestionada, el método podría adaptarse automáticamente a los cambios del problema.

c) Enfoques basados en memoria. Una forma de reutilizar en la etapa actual la información adquirida en el pasado, es almacenar en algún tipo de memoria las mejores soluciones encontradas hasta ese momento. Este enfoque funciona particularmente bien para aquellos PODs en los que los cambios siguen algún tipo de patrón periódico o repetitivo. Existen principalmente dos formas de "memorizar" las mejores soluciones encontradas: de forma explícita [18], mediante algún tipo de registro externo, o de forma implícita [28], mediante algún tipo de codificación especial de las soluciones, como los genomas diploides.

d) Enfoques basados aprendizaje o auto-adaptación. En algunos tipos de PODs, podemos encontrar patrones de cambio que siguen una cierta correlación entre ellos y que podrían permitir cierta predictibilidad [27]. En estos casos, establecer un mecanismo para "aprender" estos patrones e inferir cuáles serán los próximos cambios en el entorno, ha mostrado ser bastante efectivo de cara a acelerar el proceso de búsqueda de los nuevos óptimos en aquellos PODs que cumplen estas características. Otro enfoque basado en aprendizaje es la auto-adaptación [23]. Los métodos con auto-adaptación son aquellos que regulan o controlan ciertos parámetros de los algoritmos de optimización (tamaño de la población, probabilidad de mutación,

etc.) en base a un cierto proceso de aprendizaje que puede ser on-line (durante la búsqueda) u off-line (antes de la búsqueda). Este tipo de mecanismos permiten al método adaptarse a los cambios variando su comportamiento a través de la modificación de los correspondientes parámetros.

e) Enfoques basados en múltiples poblaciones. Otro de los enfoques propuestos para realizar un mejor seguimiento de los cambios del problema es diseñar métodos con múltiples sub-poblaciones en lugar de una única población [22]. De esta manera, se consigue una mayor diversidad y además también una mayor especialización de cada sub-población, ya que se pueden dedicar a tareas concretas como seguir un determinado óptimo, encontrar el óptimo global o rastrear los cambios en el problema.
Todos estos enfoques se han mostrado exitosos a la hora de afrontar la resolución de PODs, aunque su rendimiento tiene una gran dependencia de las características del problema que se esté abordando y también del tipo de dinamismo. Cada uno tiene sus ventajas y desventajas, y es por eso que en años recientes se ha tendido hacia una hibridación de diferentes enfoques con el objetivo de crear sinergias entre ellos y así desarrollar métodos que tengan un rendimiento más robusto cuando son aplicados a diferentes tipos de problemas.
Aunque la mayoría de los métodos de resolución propuestos para PODs están basados en poblaciones, también podemos encontrar trabajos en los que los autores abordan este tipo de problemas mediante metaheurísticas basadas en trayectorias [18], estrategias cooperativas [13, 19], portafolios de algoritmos [7], hiperheurísticas [16] o heurísticas construtivas [3]. Muchos de estos trabajos muestran como este tipo de métodos pueden obtener resultados bastante competitivos con respecto a las metaheurísticas basadas en poblaciones.

5. REVISIONES RECIENTES Y PERSPECTIVAS

En este apartado pretendemos, por un lado, ofrecer una recopilación y breve descripción de libros y artículos de revisión que se han publicado en los últimos 5 años sobre PODs. El objetivo es dirigir al lector interesado a aquellas publicaciones que le permitan tener una visión general de los avances que se han producido en este campo en las dos últimas décadas. Por otro lado, planteamos una serie de perspectivas sobre aspectos plausibles de investigar en el ámbito de los PODs.

5.1. REVISIONES

El primer trabajo al que haremos mención fue publicado en 2011 por Cruz y otros [8]. Se trata de uno de los primeros "surveys" realizados en este campo. En él, los autores revisan aspectos como los tipos de problemas abordados, concluyendo que la mayoría tratan con PODs sintéticos o con generadores de PODs, aunque se pueden encontrar algunos trabajos sobre PODs reales. También comentan los principales enfoques propuestos hasta entonces para el diseño de métodos de resolución de PODs y los principales tipos de metaheurísticas empleados. Se analizan las métricas más comúnmente usadas para la comparación y el análisis de resultados en este tipo de problemas, así como los tipos de tests estadísticos más empleados.
Un año más tarde, tres de los autores más prolíficos en este campo, T. T. Nguyen, S. Yang, y J. Branke, publicaron otra revisión del estado del arte [21], aunque esta vez centrada en el área de Computación Evolutiva (CE), probablemente la que más atención ha recibido dentro de los PODs. Este artículo recoge en primer lugar una detallada clasificación de los principales benchmarks disponibles hasta ese momento, atendiendo a diferentes criterios como número de objetivos, tipo y elementos de cambio, etc. La siguiente parte está dedicada a las medidas, donde los autores diferencian entre medidas de rendimiento, tanto para problemas mono-objetivo como multi-objetivo,y de comportamiento de los algoritmos. Tras esto, revisan las estrategias y mecanismos que se han incorporado a los métodos de CE para tratar de manera más efectiva con PODs. En el último bloque del artículo comentan los desarrollos teóricos en PODs que se habían publicado hasta ese año.
Ese mismo año también apareció otra revisión, pero en este caso orientado hacia PODs de tipo combinatorio [29]. La aparición más tardía de este survey viene a reflejar de alguna manera el hecho de que los PODs combinatorios no recibieran un mayor interés hasta fechas más recientes, en comparación con los basados en espacios de soluciones continuos. El contenido del artículo está estructurado de forma similar al que comentamos en el párrafo anterior. Los autores repasan en primer lugar los benchmarks más utilizados en la literatura relativa a PODs combinatorios, y cuáles son las diferentes características que diferencian a unos de

otros. Seguidamente exponen las medidas de rendimiento que se emplean para tipo de problemas, que como los autores comentan, son prácticamente idénticas a las usadas en los PODs continuos. Tras esto, describen los métodos de resolución propuestos para PODs combinatorios, agrupándolos según el enfoque de diseño utilizado. Finalmente, se muestra un caso de estudio sobre el problema del viajante de comercio dinámico.
En 2013 se publicaron sendos libros sobre la temática. Uno de ellos fue editado por E. Alba, A. Nakiby P. Siarry [1] con un enfoque más generalista, sin limitar la temática a un tipo concreto de métodos o de PODs. El libro consta de un total de 16 capítulos de los que la mayoría se enfocan a la aplicación de diversas clases de metaheurística a diferentes variantes de PODs, como por ejemplo, Evolución Diferencial para PODs con un número variable de óptimos, Optimización por Enjambre de Partículas para PODs multi-objetivo o Sistemas Inmunes Artificiales para PODs con restricciones, entre otros. También podemos encontrar otros capítulos que abordan aspectos metodológicos, como medidas de rendimiento o técnicas para comparación de resultados, o teóricos, como el análisis del paisaje de la función fitness. Aparte de lo anterior, encontramos capítulos que revisan la literatura relativa a al Ruteo de Vehículo Dinámico o a los PODs temporalmente-dependientes (time-linkage), en los que la solución escogida en la etapa actual puede influir en los cambios futuros del problema.
El otro libro [32], editado en este caso por S. Yang y X. Yao, se centra de nuevo en el área de la Computación Evolutiva (CE). Este se divide en cuatro bloques. En el primero tiene un enfoque más introductorio, y consta de cuatro capítulos en los que se dan algunas nociones básicas sobre DOPs (definición, benchmarks disponibles y medidas de rendimiento), se revisan las estrategias de CE desarrolladas para PODs tanto mono-objetivo como multi-objetivo, y se realiza un análisis crítico de la investigación hecha hasta entonces en este campo. La siguiente parte del libro se focaliza en el diseño de diversas metodologías de CE para resolución de PODs tanto con restricciones como sin restricciones. El tercer bloque aborda aspectos teóricos en PODs, con capítulos donde se revisan los trabajos publicados en dicha temática y la aplicación del análisis del paisaje de la función objetivo, o se analiza teóricamente el comportamiento de los algoritmos evolutivos en entornos dinámicos mono y multi-objetivo. La última parte del libro está dedicada a las aplicaciones reales. Aquí el lector interesado encontrará varios trabajos sobre casos prácticos de PODs en áreas como el problema del viajante de comercio dinámico, la redes móviles ad-hoc, el manejo de drones, o la optimización de cadenas de suministro.
Jordehi en [25], presentó en 2014 otro interesante artículo de revisión enfocado a un tipo concreto de métodos que también han dado muy buenos resultados en PODs, la Optimización por Enjambre de Partículas (PSO, por sus siglas en inglés). El autor comienza detallando los retos que han de abordar los algoritmos basado en PSO para resolver PODs, así como los benchmarks y medidas de rendimiento más comunes para este tipo de métodos. Luego se ofrece una exhaustiva revisión de las diferentes variantes de los métodos PSO para PODs que podemos encontrar en la literatura, clasificándolos según el enfoque seguido por los investigadores a la hora de adaptar su funcionamiento a las singularidades y requerimientos de este tipo de problemas.
Más recientemente, ha aparecido otro interesante survey [23] orientado a un área que está acaparando una gran atención durante los últimos años en el campo de los PODs, los métodos de resolución auto-adaptativos. Novoa-Hernández y co-autores revisan en primer lugar los trabajos más relevantes publicados sobre esta temática, y proponen una clasificación en tres categorías dependiendo de donde se localiza la estrategia de auto-adaptación: en el propio método de optimización, en el mecanismo diseñado para seguir los cambios en el problema, o en ambos. En la segunda parte del artículo, usando el algoritmo de Evolución Diferencial como referencia, realizan un estudio empírico del papel que juega la auto-adaptación en este tipo de entornos dependiendo del tipo de método considerado, de acuerdo a la anterior clasificación.

5.2. PERSPECTIVAS

Una vez introducidos los aspectos esenciales de los PODs y comentado las revisiones más recientes sobre el tema, en esta sección indicamos cuáles, en nuestra opinión, son temas pendientes y relevantes en la investigación asociada con PODs.

Coste de los cambios: en general, se considera que la adaptación de la solución al cambio producido es una operación "libre de coste". Sin embargo, esto es una suposición falsa en problemas reales. La inclusión de un coste de adaptación lleva directamente a un problema bi-objetivo: (maximizar beneficio, minimizar coste de adaptación) que debe ser tratado con las herramientas adecuadas.

Test estadísticos: la validación estadística de los resultados obtenidos en PODs sigue un enfoque muy similar al realizado en problemas estáticos, teniéndose únicamente en cuenta el resultado final. Aunque se han propuesto medidas de rendimiento, como el error off-line, que intentan agregar el rendimiento de las metaheurísticas a lo largo de toda la ejecución, basar el análisis de significación estadística en un único valor, cuando estamos considerando entornos dinámicos, puede dar lugar a importantes sesgos en la interpretación de los resultados ofrecidos por los test estadísticos. El empleo de tests que permitan determinar diferencias significativas en la convergencia de los algoritmos, como el presentado en [11], puede ser una alternativa interesante a explorar.

Sobre los "benchmarks": al igual que ocurre en la investigación en metaheurísticas para optimización, los investigadores recurren (recurrimos) a generadores de problemas o problemas de prueba que brindan un entorno controlado de experimentación. Sin embargo, intuimos que estas herramientas pueden inducir a estudiar escenarios que no tengan una contraparte en el mundo "real" (por ejemplo, respecto al tipo de cambio esperado). Es necesaria la definición y creación de librerías de PODs "realistas" que permitan reducir la brecha entre la investigación puramente académica y las aplicaciones reales.

Tipos de cambio: la mayoría de los estudios consideran que los cambios se producen en la función de coste. Sin embargo, y si pensamos sobre todo en problemas de tipo combinatorio, es razonable tener en cuenta cambios en otros aspectos del problema (por ejemplo, la demanda generada en un problema de localización, el tráfico o la cantidad de ciudades en el problema de viajante de comercio, etc.)

Métodos de resolución: el uso de métodos evolutivos o basados en poblaciones es claramente dominante en la investigación sobre PODs. La razón subyacente tiene que ver con la relación entre la "evolución" y su capacidad para adaptarse a los cambios. Sin embargo, en el contexto de los PODs ha habido poca exploración en el uso de otras técnicas de optimización, del tipo búsqueda local o métodos constructivos. Estas últimas pueden jugar un rol clave puesto que no es claro si en los PODs resulta mejor "buscar" una solución o "construir" una nueva a partir de la ya disponible.

¿Cuándo se cambia la solución?: un aspecto relevante a estudiar es en qué momento se "implanta" o "despliega" la nueva solución encontrada después de un cambio. En otras palabras, cuándo se desecha la configuración actual y se reemplaza por la nueva. Según nuestro conocimiento, ésto no ha sido considerado.

PODs multiobjetivo: la investigación realizada hasta ahora en PODs se ha centrado principalmente en problemas mono-objetivo. Sin embargo, esta característica es difícil de encontrar en entornos reales, en los que habitualmente hay que optimizar varios objetivos contradictorios entre sí. Si bien en los últimos años los PODs multi-objetivo han recibido una mayor atención, todavía queda mucho camino por recorrer teniendo en cuenta que la complejidad y variabilidad de estos problemas es bastante mayor que la de sus contrapartes mono-objetivo.

AGRADECIMIENTOS

D. Pelta agradece el apoyo del Ministerio de Economía y Competitividad (España) y de la Consejería de Innovación, Ciencia y Empresa de la Junta de Andalucía, a través de los proyectos TIN2014-55024-P y P11-TIC-8001 (ambos incluyen Fondos FEDER).
A. D. Masegosa agradece el apoyo del Ministerio de Economía y Competitividad de España a través de los proyectos TEC2013-45585-C2-2-R y TIN2014-56042-JIN.

RECEIVED: JULY, 2016
REVISED: NOVEMBER, 2016

REFERENCIAS

[1] ALBA, E., NAKIB, A., and SIARRY, P., editors [2013]: **Metaheuristics for Dynamic Optimization**, volume 433 of **Studies in Computational Intelligence** Springer Berlin Heidelberg, Berlin, Heidelberg.

[2] AMO, I. G. D., PELTA, D. A., GONZÁLEZ, J. R., and MASEGOSA, A. D. [2012]: An algorithm comparison for dynamic optimization problems **Applied Soft Computing**, 12, 10,:3176 – 3192.

[3] Baykasoglu, A. and Ozsoydan, F. B. [2015]: A constructive search algorithm for combinatorial dynamic optimization problems In **Proceedings of the 2015 IEEE International Conference on Evolving and Adaptive Intelligent Systems (EAIS)**, pages 1–7.

[4] BEASLEY, J. E. [1990]: Or-library: distributing test problems by electronic mail **Journal of the Operational Research Society**, 41, 11,:1069–1072 http://people.brunel.ac.uk/ mastjjb/jeb/info.html.

[5] BEN-ROMDHANE, H., ALBA, E., and KRICHEN, S. [2013]: Best practices in measuring algorithm performance for dynamic optimization problems **Soft Computing**, 17, 6,:1005–1017.

[6] BRANKE, J. [1999]: Memory enhanced evolutionary algorithms for changing optimization problems In **Proceedings of the 1999 IEEE Congress on Evolutionary Computation (CEC-1999)**, pages 1875–1882.

[7] CALDERÍN, J. F., MASEGOSA, A. D., and PELTA, D. A. [2015]: Algorithm portfolio based scheme for dynamic optimization problems **International Journal of Computational Intelligence Systems**, 8, 4,:667–689.

[8] CRUZ, C., GONZÁLEZ, J., and PELTA, D. [2010]: Optimization in dynamic environments: a survey on problems, methods and measures **Soft Computing**, 15, 17,:1–22.

[9] del Amo, I. G., Pelta, D. A., and González, J. R. [2010]: Using heuristic rules to enhance a multiswarm PSO for dynamic environments In **Proceedings of the 2010 IEEE Congress on Evolutionary Computation**, pages 1–8.

[10] DEL AMO, IGNACIO G. AND PELTA, DAVID A. [2013]: SRCS: A technique for comparing multiple algorithms under several factors in dynamic optimization problems In [1], pages 61–77.

[11] DERRAC, J., GARCÍA, S., HUI, S., SUGANTHAN, P. N., and HERRERA, F. [2014]: Analyzing convergence performance of evolutionary algorithms: a statistical approach **Information Sciences**, 289:41–58.

[12] GABREL, V., MURAT, C., and THIELE, A. [2014]: Recent advances in robust optimization: An overview **European Journal of Operational Research**, 235, 3,:471 – 483.

[13] GONZÁLEZ, J. R., MASEGOSA, A. D., and GARCÍA, I. J. [2011]: A cooperative strategy for solving dynamic optimization problems **Memetic Computing**, 3, 1,:3–14.

[14] JIANG, S. and YANG, S. [2016. In press]: Evolutionary Dynamic Multiobjective Optimization: Benchmarks and Algorithm Comparisons **IEEE Transactions on Cybernetics**.

[15] JIN, Y. and BRANKE, J. [2005]: Evolutionary optimization in uncertain environments-a survey **IEEE Transactions on Evolutionary Computation**, 9, 3,:303–317.

[16] KIRAZ, B., ETANER-UYAR, A., and ÖZCAN, E. [2013]: Selection hyper-heuristics in dynamic environments **Journal of the Operational Research Society**, 64, 12,:1753–1769.

[17] LI, C., YANG, S., and PELTA, D. A. [2011]: Benchmark generator for the IEEE WCCI-2012 competition on evolutionary computation for dynamic optimization problems Technical report, China University of Geosciences, Brunel University, University of Granada.

[18] MASEGOSA, A. D., ONIEVA, E., LOPEZ-GARCÍA, P., OSABA, E., and PERALLOS, A. [2015]: An adaptive local search with prioritized tracking for Dynamic Environments **International Journal of Computational Intelligence Systems**, 8, 1,:1053–1075.

[19] MASEGOSA, A. D., PELTA, D., and DEL AMO, I. G. [2014]: The role of cardinality and neighborhood sampling strategy in agent-based cooperative strategies for Dynamic Optimization Problems **Applied Soft Computing**, 14:577–593.

[20] NAKIB, A. and SIARRY, P. [2013]: Performance analysis of dynamic optimization algorithms In [1], pages 1–16.

[21] NGUYEN, T. T., YANG, S., and BRANKE, J. [2012]: Evolutionary dynamic optimization: A survey of the state of the art **Swarm and Evolutionary Computation**, 6:1–24.

[22] NOVOA-HERNÁNDEZ, P., CORONA, C. C., and PELTA, D. A. [2011]: Efficient multi-swarm PSO algorithms for dynamic environments **Memetic Computing**, 3, 3,:163–174.

[23] NOVOA-HERNÁNDEZ, P., CORONA, C. C., and PELTA, D. A. [2016]: Self-adaptation in dynamic environments - a survey and open issues **International Journal of Bio-Inspired Computation**, 8, 1,:1–13.

[24] REINELT, G. [1995]: TSPLIB https://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/.

[25] REZAEE JORDEHI, A. [2014]: Particle swarm optimisation for dynamic optimisation problems: a review **Neural Computing and Applications**, 25, 7-8,:1507–1516.

[26] RICHTER, H. [2009]: Detecting change in dynamic fitness landscapes In **Proceedings of the 2009 IEEE Congress on Evolutionary Computation**, pages 1613–1620.

[27] SIMÕES, A. and COSTA, E. [2014]: Prediction in evolutionary algorithms for dynamic environments **Soft Computing**, 18, 8,:1471–1497.

[28] YANG, S. [2006]: On the design of diploid genetic algorithms for problem optimization in dynamic environments In **Proceedings of the 2006 IEEE International Conference on Evolutionary Computation**, pages 1362–1369.

[29] YANG, S., JIANG, Y., and NGUYEN, T. T. [2012]: Metaheuristics for dynamic combinatorial optimization problems **IMA Journal of Management Mathematics**, 24, 4,:451–480.

[30] YANG, S. and YAO, X. [2005]: Experimental study on population-based incremental learning algorithms for dynamic optimization problems **Soft Computing**, 9, 11,:815–834.

[31] YANG, S. and YAO, X. [2008]: Population-based incremental learning with associative memory for dynamic environments **IEEE Transactions on Evolutionary Computation**, 12, 5,:542–561.

[32] YANG, S. and YAO, X., editors [2013]: **Evolutionary Computation for Dynamic Optimization Problems**, volume 490 of **Studies in Computational Intelligence** Springer Berlin.